Présentation des Données du Web TP 2 : XPath et XQuery - Partie 2
Federico Ulliana

TP en binômes à rendre le 11 octobre. Le rendu est composé de deux parties : i) le present TP et ii) celui du 23 septembre. Les deux sont à rendre au même moment (11 octobre), dans le même document. Un seul document pdf (synthétique,max 5MB) par binôme est demandé, et sera à déposer sur Moodle.

1) Warm-up BaseX

On utilisera le logiciel BaseX pour interroger des documents XML avec des requêtes XQuery.

2) XQuery sur les Tweets (avec BaseX)

Reprenez votre DTD pour les Tweets, et créez un document valide par rapport à votre schéma contenant au moins 3 utilisateurs et 5 tweets. Attention : il sera peut être nécessaire d'apporter des légères modifications à votre DTD afin de permettre des interrogations !

Donnez les requêtes XQuery correspondants aux expressions suivantes :

1. Créez une liste de paires tweet-auteur, avec chaque paire contenue dans un element `result`.
2. Pour chaque utilisateur, listez la date de tous ses tweets groupés dans un élément `result`.
3. Listez les utilisateurs qui ont publié un tweet qui a été retwitté au moins deux fois.
4. Pour chaque tweet, listez son contenu et la date de ses deux premiers retweets. Rajoutez un element vide `<nonRetwitted/>` s'il n'a pas été retwitté.
5. Listez les utilisateurs de la plateforme en ordre alphabétique.
6. Listez les tweets contenants l'hashtag #I<3XML.
7. Trouvez le tweet le plus ancien ainsi que le plus recent.
8. Pour chaque tweet ayant des hashtags, retournez le tweet avec la liste de ses hashtag.
9. Pour chaque tweet ayant des références utilisateur, retournez le tweet avec la liste des références utilisateur.
10. Déclarez la fonction `fn:retwittePar`, qui, étant donné un tweet, retourne tous les utilisateurs qui l'ont retwitté.

3) Génération de Pages HTML via XQuery

Sur la base des données à l'adresse http://opendata.montpelliernumerique.fr/datastore/VilleMTP_MTP_ZATAntigone_2011.xml, écrire un programme XQuery permettant de générer une page HTML contenant trois sections qui présentent les spectacles de la ville de Montpelier regroupes à la fois par (i) date, (ii) titre, et (iii) auteur. Discutez votre choix de gestion des auteurs en cas de valeurs manquants. La fonction XQuery `doc(url)` permet la récupération de données distantes.

4) Propriétés des requêtes XPath et XQuery

Reformuler les requêtes suivantes en utilisant exclusivement les axes `child`, `descendant`, `descendant-or-self`, `following` et `following-sibling`

- `//b[parent::a]`
- `//a/preceding-sibling::c`
- `//c[preceding::d]`
- `//b/a/preceding-sibling::c/preceding::d`

— `/a/b/..//*/./../preceding::d`
— `//a/ancestor::b/parent::c/child::d/parent::e`

1. Pour chaque requête, proposer un document XML pour lequel la réponse à la requête n'est pas vide, sinon expliquer pourquoi un tel document n'existe pas.
2. Pour chaque requête, dire si elle est non-vide pour quelque document valide par rapport à la DTD `<!DOCTYPE a [<!ELEMENT a (c)> <!ELEMENT c ANY>]>`
3. Pour le langage XPath, est il vrai que si $X = Y$ et $Y = Z$ alors $X = Z$? Est cela aussi le cas pour XQuery ?
4. Donner un document XML pour lequel la requête $//\mathtt{r}[\mathtt{a}[1] = \mathtt{a}[2]]$ n'est pas vide.
5. Proposez trois requêtes XPath permettant de sélectionner la racine "`r`" après avoir testé la valeur des attributs des noeuds étiquetés avec "`a`". `<r> <a id="c" ref="c"></a> <a ref="a" id="c"></a> </r>`